View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Dokumenten-Publikationsserver der Humboldt-Universität zu Berlin

"Die Suche nach der Nadel im Heuhaufen" - Nyx - Ein System zur Lokalisierung von Rechnern in großen Netzwerken anhand IP- oder MAC-Adressen

Ralf Kornberger, Helmut Reiser

Leibniz-Rechenzentrum, Garching bei München

{kornberger, reiser}@lrz.de

Abstract: Diese Arbeit präsentiert eine Lösung zur Lokalisierung von Endgeräten in großen Netzwerken. Unter Lokalisierung ist die Identifikation der Netzkomponente am Netzrand (Edge-Switch), sowie der Anschlussport des Endgerätes an diesem zu verstehen. Die Herausforderung war die Erstellung eines Datenmodells mit Verknüpfung von IP- und MAC-Adresse, sowie des Anschlussports am Edge-Switch. In einem großen und dynamischen Netzwerk ist eine einmalige Erfassung und statische Speicherung mit periodischer Aktualisierung nicht ausreichend. Deshalb wurde ein hochparallelisiertes System entwickelt, das die Daten fortlaufend aktualisiert und mit Hilfe maschinellen Lernens auch mit dynamischen, komplexen und heterogenen Topologien zurecht kommt.

1 Einleitung

Betreibt man als Provider ein großes Netzwerk mit mehreren zehntausend Benutzern und Endgeräten, sieht man sich mit einer sehr komplexen Infrastruktur konfrontiert. Als zentraler Ansprechpartner für alle seine Netzbereiche und über 1500 Subnetze ist das Leibniz-Rechenzentrum (LRZ) verantwortlich für die Bearbeitung von Abuse-Fällen. Potentiell kompromittierte Rechner müssen ausfindig gemacht (lokalisiert) werden, damit die lokalen Administratoren oder das Rechenzentrum selbst Gegenmaßnahmen ergreifen können, also z.B. den Anschluss der betroffenen Rechner zu sperren. Das gilt auch für Sicherheitsverstöße, z.B. exessive Port-Scans, die von den eigenen Monitoring-Systemen bzw. Intrusion Detection Systemen (IDS) erkannt werden, oder in Fällen, bei denen die Benutzer gegen die Benutzungsrichtlinen verstoßen.

Das Münchner Wissenschaftsnetz (MWN [LR06]) wird vom LRZ für die beiden Münchner Universitäten, verschiedene Fachhochschulen und Instituten betrieben. Das Netz verteilt sich auf über 60 Standorte mit ca. 55.000 angeschlossenen Endgeräten[1] und kann von allen Studenten und wissenschaftlichen Mitarbeitern der angebundenen Hochschulen genutzt werden, d.h., dass potentiell 100.000 Menschen das MWN verwenden. Das LRZ stellt die Netzwerkinfrastruktur zur Verfügung und hat keine administrativen Rechte an den angeschlossenen Endsystemen. Diese werden von lokalen Administratoren betreut, die ihrerseits keine globale Sicht auf alle Endgeräte haben und nur ihren eigenen Bereich betreuen. Im MWN gibt es über 700 lokale unanhängige Administratoren, die Institute, Lehrstühle und Studentenwohnheime betreuen.

[1] ohne "Wireless"-Geräte

1.1 Szenario

Die Lokalisierung von IP- und MAC-Adressen ist immer dann notwendig, wenn der genaue Standort eines Endgerätes von Bedeutung ist. Das kann im Fall von Wartungsarbeiten oder Störungen sein. Am häufigsten aber tritt die Notwendigkeit der Lokalisierung eines Endgerätes in Zusammenhang mit Abuse-Fällen auf. Bei IP-Adressen, die durch ein ungewöhnlich hohes Datenverkehrsvolumen auffallen (z.B. durch Versand von SPAM-Emails) oder von automatischen Monitoring- und Intrusion-Detection-Systemen erkannt werden, muss das zur IP-Adresse gehörende Endgerät eindeutig identifiziert werden, damit das Sicherheitsproblem behoben werden kann. Das kann Verkehr zu verdächtigen Servern im Internet, sog. “command and control”-Server von Bot-Netzen, sein oder eine Kommunikation über bekannte Ports von Trojanern. Zwar ist es möglich, die betroffen IP-Adresse am Edge-Router zu sperren, was aber keine dauerhafte Lösung sein soll. So können die gesperrten Rechner zwar nicht mehr mit dem Internet kommunizieren, stellen aber weiterhin eine Gefahr für Rechner innerhalb des MWN da. Außerdem ist eine dauerhafte Sperrung bei dynamisch vergebenen IP-Adresse sinnlos. Ziel muss es sein, das Problem auf dem betroffenen Endgerät schnell zu beseitigen, durch Update bzw. Installation eines Virenscanners oder durch komplette Neuinstallation des Betriebssystems, damit es wieder seinen normalen Betrieb aufnehmen kann. Die eindeutige Identifizierung geschieht mit Hilfe der IP- und MAC-Adresse, die nun im Netz lokalisiert werden müssen. Im Sinne dieser Arbeit ist unter Lokalisierung die Identifikation der Netzkomponente am Netzrand (Edge-Switch), sowie der Anschlussport des Endgerätes am Edge-Switch zu verstehen. Die hier vorgeschlagene Lokalisierungslösung ”Nyx”[2] verfolgt nun das Ziel, die Suche nach einem bestimmten Endgerät zu automatisieren, zu optimieren und zu zentralisieren.

1.2 Problemstellung / Anforderungen

In einem großen und dynamischen Netzwerk, wie dem Münchner Wissenschaftsnetz, das über 1000 Netzkomponenten enthält, ist es keine triviale Aufgabe den Standort eines bestimmten Endgerätes zu ermitteln. Das Netzwerk unterliegt ständigen Veränderungen. Zum einen wird die Netzwerkinfrastruktur fortlaufend aus- und umgebaut, sowie modernisiert. Zum anderen verändern sich auch die Standorte der Endgeräte. Es kommen ständig neue hinzu und alte fallen weg. Außerdem werden für große Adressbereiche die IP-Adressen dynamisch vergeben und wechseln deshalb auch für ansonsten stationäre Geräte. In Anbetracht dessen sieht man sich mit einem hoch dynamischen System konfrontiert, für das eine einmalige Erfassung und statische Speicherung mit periodischer Aktualisierung nicht ausreicht. Sicherheitsvorfälle aus der Praxis haben gezeigt, dass der Bestand der Daten aus den Netzkomponenten ständig, d.h. im Bereich von Minuten bis maximal einer halben Stunde (Aging-Time der MAC-Forwarding-Tabellen), aktualisiert werden muss. Ansonsten können wichtige Daten verloren gehen, sodass einige Endgeräte nicht erfasst werden. Dieser Prozess des Datensammelns sollte parallelisiert ablaufen, um das Kriterium der Echtzeit erfüllen zu können.

Um die gewonnenen Daten filtern und verdichten zu können, ist die Kenntnis der Topologie der Netzkomponenten zwingend notwendig. Eine besondere Herausforderung dafür ist die heterogene Struktur der Netzkomponenten im MWN. Nicht alle Geräte unterstützen standardisierte Topologieerkennungsprotokolle, so dass eine einfache automatisierte Erkennung der Topologie nicht möglich ist. Es muss daher ein anderer Weg gefunden werden, die Topologie maschinell möglichst zuverlässig, d.h. mit hoher Genauigkeit, zu erkennen. Manuelle Ansätze scheitern im MWN am unverhältnismäßigen Aufwand, vor allem aber an den harten Echtzeitanforderungen.

[2] in der griechischen Mythologie die personifizierte Nacht

1.3 Lösungsansatz

Um die Informationen über die Anschlussports der Endgeräte in Echtzeit zur Verfügung stellen zu können, werden fortlaufend sowie hoch parallelisiert die benötigten Daten von den Netzkomponenten ermittelt und in einer zentralen Datenbank gespeichert. Zusätzlich enthält die Datenbank eine History, damit auch ältere Standorte abfragbar sind.

Zur Topologieerkennung wird ein Ansatz aus dem Bereich des maschinellen Lernens verwendet. Statt der konkreten Topologie werden nur die einzelnen Verbindungen zwischen den Netzkomponenten erkannt: die Up- und Downlinkports[3]. Eine Kenntnis der vollständigen Topologie ist für die Erkennung von Edge-Switch-Ports nicht notwendig.

1.4 Aufbau der Arbeit

Kapitel 2 geht zunächst auf bekannte Verfahren zur Topologieerkennung ein. Danach wird im dritten Kapitel Netdisco, ein System mit ähnlicher Funktionsweise wie Nyx, kurz vorgestellt. In Kapitel 4 folgt dann eine detailierte Beschreibung der Problemstellung und des Designs des am LRZ entwickelten Systems. Dazu gehören das Datenmodell, eine neue Form der Topologieerkennung mit der letztendlichen Zuordnung einer IP-Adresse zu einem Switch-Port und das Parallelisierungskonzept der Software. Kapitel 5 geht dann auf die eigentliche Implementierung der Software "Nyx" ein. Am Ende der Arbeit in Kapitel 6 werden Erfahrungen aus dem praktischen Betrieb geschildert.

2 Topologieerkennung

Zur Topologieerkennung existieren mehrere Protokolle und Verfahren. Die zwei wichtigsten Protokolle, sowie andere Ansätze werden hier kurz vorgestellt und bewertet.

2.1 Cisco Discovery Protocol (CDP)

Das Cisco Discovery Protocol (CDP [Sys]) ist ein von Cisco Systems entwickeltes proprietäres Protokoll. Es dient Netzkomponenten dazu, sich untereinander mit der sog. "neighbour discovery" zu finden. Dazu werden in regelmäßigen Abständen Ethernet-Frames ("advertisements") an die Multicast-MAC-Adresse 01:00:0C:CC:CC:CC auf allen für CDP aktivierten Ports gesendet, mit denen das Gerät sich selbst gegenüber anderen Geräten ankündigt. Außerdem erwartet es auf dieser Multicast-MAC-Adresse Advertisements anderer Geräte. Somit sieht eine CDP-fähige Netzkomponente alle ihre CDP-fähigen Nachbarn. Die Ports mit CDP-fähigen Nachbarn sind Up-/Downlinkports. CDP ist hauptsächlich auf Netzkomponenten von Cisco in Einsatz.

Einige Komponenten andere Hersteller unterstützen CDP allerdings nur passiv. Sie sind in der Lage Advertisements auszuwerten, senden aber ihrerseits keine eigenen. Deshalb ist CDP in heterogenen Netzen wie dem MWN nur von geringem Nutzen, da nur sehr wenige Komponenten CDP voll unterstützen.

[3] ab jetzt als Up-/Downlinkports bezeichnet, da die Datenflussrichting für Nyx irrelevant ist

2.2 Link Layer Discovey Protocol (LLDP)

Das Link Layer Discovey Protocol (LLDP) ist ein neuer offener herstellerunabhängiger IEEE-Standard [IEE05]. Es hat ähnliche Aufgaben und Funktionen wie CDP, ist dazu aber inkompatibel. Die Multicast-MAC-Adresse von LLDP lautet
01:80:c2:00:00:0e. Alle Ports, auf denen LLDP-fähige Geräte entdeckt werden, sind in der LLDP-basierten Topologieerkennung Up-/Downlinkports. LLDP hat den Vorteil, dass es wegen seiner Offenheit von vielen Herstellern unterstützt werden kann. Cisco z. B. unterstützt aber weitgehend noch kein LLDP.

Im MWN wird LLDP von vielen Netzkomponenten unterstützt. Deshalb können die so gewonnenen Informationen sinnvoll weiterverwertet werden. Trotzdem gibt es aber Lücken durch LLDP-inkompatible Geräte, sodass LLDP alleine zur Topologieerkennung nicht ausreicht.

2.3 Andere Methoden

Außer den beiden oben vorgestellten Methoden, gibt es noch die Möglichkeit, die Topologie mit manuellen Verfahren oder Heuristiken zu erkennen. Sie kommen dann zum Einsatz wenn CDP und LLDP gar nicht oder nur eingeschränkt zur Verfügung stehen. Bei einer Methode wird die Netztopologie von Hand gepflegt, entweder mit einer Software direkt oder in einer externen Datenbank. Große Nachteile dieser Methode sind der hohe Arbeitsaufwand zur Erstellung und Pflege der Daten sowie häufig deren geringe Aktualität. Für kleine Netze ist diese Methode aber durchaus brauchbar.

Für Netze mit geringer bis mittlerer Komplexität kann die heuristische Methode eine brauchbare Lösung sein. Die Heuristik setzt an der Stelle an, an der CDP und LLDP nicht weiterhelfen, weil z.B. eine verwendetet Netzkomponente nur CDP kann während alle anderen nur LLDP können. Man versucht dann mit Algorithmen die durch CDP bzw. LLDP erkannten Teilnetze durch logische Schlüsse und Datenabgleiche zusammenzuführen, z.B. über Korrelation von IP-Subnetzen. Wird das Spanning-Tree-Protokoll flächendeckend eingesetzt, ist es ebenfalls möglich, daraus mittels gelernter MAC-Adressen Schlüsse auf die Topologie zu ziehen [BGJ+04].

Das MWN ist aber so groß, dass manuelle Erfassung nur in großen Zeitabständen möglich ist und die Informationen nicht tagesaktuell sind. Heuristische Methoden liefern wegen der Komplexität des MWN nur bedingt brauchbare Ergebnisse, d.h. sie sind nur in Teilbereichen einsetzbar. Spanning-Tree wird auch nur in sehr wenigen Subnetzen verwendet.

3 Verwandte Systeme: Netdisco

Neben dem in diesem Papier vorgestellten Nyx gibt es noch andere quelloffene (“open-source”) Systeme, die einen ähnlichen Funktionsumfang wie Nyx bieten, sich in Details aber stark davon unterscheiden. Netdisco ist vom Funktionsumfang am ehesten mit Nyx zu vergleichen.

Netdisco [MFB] ist ein webbasiertes Werkzeug zum Netzwerkmanagement. Es ist für mittlere bis große Netze ausgelegt. Die Daten der Switches und Router werden wie bei Nyx (siehe Kapitel 5) regelmäßig per Managementprotokoll abgefragt und in einer SQL-Datenbank gespeichert. Die Datenbank enthält auch eine zeitlichen Verlauf, der über alte Strandorte von Endgeräten Auskunft gibt. Außerdem ist es möglich Switch-Ports über die Weboberfläche zu verwalten, z.B. ein- oder auszuschalten. Die in Netdisco integrierte To-

pologieerkennung stützt sich auf CDP, Foundry Discovery Protocol sowie weitere proprietäre Protokolle, und findet so auch automatisch neue Netzkomponenten. Netdisco wurde in Perl programmiert und ist unter FreeBSD, Linux und Solaris ausführbar.

Netdisco wurde vom LRZ vor der Entwicklung von Nyx getestet. Allerdings war es für das MWN ungeeignet, da alle Netzkomponenten seriell statt parallel (siehe Kapitel 4.3) abgefragt wurden. Somit konnte die Echtzeitbedingung des LRZs nicht erfüllt werden, da eine serielle Abfrage aller Netzkomponenten im MWN mehr als zwei Stunden benötigt. Netdisco diente Nyx aber als Vorbild.

4 Design

Nyx wurde so entworfen, dass alle Aufgaben vollständig automatisiert ablaufen. Manuelle Eingriffe sind nur zur Erstellung der Startkonfiguration vorgesehen. Ein wichtiges Konzept, das vor allem bei der Implementierung der Software (vgl. Abschnitt 5.1) eine große Rolle spielt, ist die Aufteilung der einzelnen Aufgaben in disjunkte Teilbereiche, um eine möglichst hohe Parallelisierung erreichen zu können. Im folgenden werden die drei wichtigsten Basiskonzepte vorgestellt.

4.1 Topologieerkennung

In sehr großen Netzwerken wie dem MWN ist eine grobe Eingrenzung zwar anhand der IP-Adresse möglich, da an verschiedene Institutionen unterschiedliche IP-Adressbereiche vergeben wurden. Im MWN gibt es aber Institutionen, die einen Netzbereich für mehr als 60000 Endgeräte (Klasse-B-Netz) besitzen.

Eine Eingrenzung auf ein bestimmtes Institut, Departement oder einen Lehrstuhl ist hier nicht mehr möglich, falls nur die MAC-Adresse bekannt ist. Daher müssten im schlimmsten Fall alle Netzkomponenten durchsucht werden. In Netzwerken wie dem MWN können das mehrere hundert Netzkomponenten sein.

Die MAC-Adressen der Endgeräte sind nicht nur am Edge-Switch und am ersten Router sichtbar, sondern auch an allen an der Kommunikationskette beteiligten Core-Switches (siehe Abbildung 1), sodass es ohne Hilfswerkzeug sehr aufwändig wäre, den Switch zu finden, an dem ein bestimmtes Endgerät angeschlossen ist. Dazu müsste die Kommunikationskette manuell zurückverfolgt werden.

Für eine Lokalisierung muss der Edge-Switch ausfindig gemacht werden, an dem das gesuchte Endgerät angeschlossen ist. Um dieses zu erreichen müssen die überflüssigen Daten vom Endgerät, die an allen Netzkomponenten zwischen Router und Edge-Switch anfallen, herausgefiltert werden, so dass nur noch die Daten, die vom Edge-Switch abgefragt werden, in der Datenbank enthalten sind. Würde nicht gefiltert, würde eine Flut von Daten in die Datenbank übernommen. Die MAC-Adresse des Endgerätes ist pro Switch jeweils am Downlinkport sichtbar. Die MAC-Adresse des PCs würde in der Beispiel-Topologie vier mal (drei mal Switch Downlinks + einmal Router) auftauchen, ohne dass der Edge-Switch besonders hervorsticht. Ziel der Topologieerkennung in Nyx ist es, die Ports der Zwischenstationen vom Port des Endgerätes am Edge-Switch zu unterscheiden. Handelt es sich um den Edge-Switch, so ist der Downlinkport ein einfacher Datenport, an dem ein Endgerät, und keine weitere aktive Netzkomponente, angeschlossen ist. Diese Unterscheidung kann aber ohne weitere Informationen nicht getroffen werden. Dazu werden Topologie- oder vergleichbare Informationen benötigt.

Da die Kenntnis der genauen Topologie nicht zwingend notwendig ist, wird hier gegen-

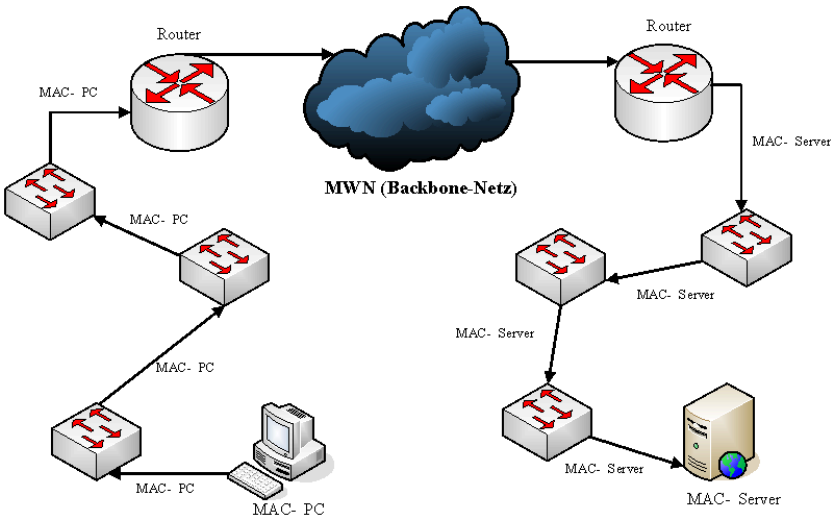

Abbildung 1: Beispiel-Topologie: die MACs der Endgeräte sind an jedem Switch bishin zum Router sichtbar

Interface ID	#MACs	%Traffic	SW/RT MAC	CDP/LLDP	Tagged	Up/Downlink
100007	491	13	Ja	Ja	Ja	Ja
200002	6	17	Nein	Ja	Ja	Ja
300017	1	33	Nein	Nein	Nein	Nein
400023	4	25	Nein	Nein	Ja	Nein *
500010	80	45	Ja	Nein	Nein	Ja
600008	52	22	Nein	Nein	Nein	Nein **
700009	20	1	Nein	Nein	Ja	Ja

Tabelle 1: beispielhafter Ausschnitt: Verkehrsdaten der Switchports, *=Server mit Tagging, **=nicht verwaltbarer Switch

über oben erwähnten Verfahren (vgl. Kapitel 2) ein neuer Ansatz gewählt. Statt strikter Topologieerkennung verwendet Nyx Mustererkennung, um die Ports der Switches zu klassifizieren, die zum Router (Uplinkport), einem weiteren Switch (Downlinkport) oder einem Endgerät (Datenport) führen. Die Information welche Switches direkt miteinander verbunden sind, ist für die Lokalisierung allerdings irrelevant. Deshalb handelt es sich um keine klassische Topologieerkennung. Für die Mustererkennung wird ein maschineller Lernalgorithmus verwendet. Dieser muss aber zu Beginn mit sog. Seed-Daten trainiert werden (”supervised learning”). Die Seed-Daten werden einmal manuell erstellt. Als Datengrundlage dienen Verkehrsdaten, die pro Anschlussport eines Switches erhoben werden, beim LRZ für ca. 50000 Anschlussports.

Die Verkehrsdaten charakterisieren das Verhaltensmuster der Geräte, die an diesem Switchport angeschlossen sind. Ein Teil dieser Daten besteht aus der Anzahl der MAC-Adressen am Port, sowie dem Prozentsatz des Ports am gesamten Datenverkehr des Switches. Daraus lässt sich ableiten, ob ein Port ein Up-/Downlinkport ist, da eine große Anzahl von MAC-Adressen normalerweise nicht nur zu einem einzelnen Endgerät gehört, ebenso wenig wie ein hoher Verkehrsanteil. Eine Außnahme bilden nicht verwaltbare Switches, die hier als Endgeräte betrachtet werden.

Da diese Kriterien aber zur Entscheidungsfindung nicht ausreichen, wird weiterhin ermittelt, ob an dem Port die MAC-Adresse einer bekannte Netzkomponente gesehen wurde. Ist dieses der Fall, handelt es sich um einen Up-/Downlinkport, da über ihn ein anderer

Switch oder Router direkt zu Verwaltungszwecken (z.B. Managementabfragen) angesprochen wurde, Spanning-Tree aktiviert ist oder im Falle eines Router man dessen MAC-Adresse beim Verkehr zurück zum Endgerät sieht. Zu den Verkehrsdaten zählt auch die Information, ob über LLDP eine benachbarte Netzkomponente entdeckt wurde, was ebenfalls auf einen Up-/Downlinkport bei positiven Ergebnis schließen lässt. Das gilt auch für markierte (”tagged”) Ports, da nur selten solche Ports für Endgeräte verwendet werden.

Diese Kriterien werden aber nicht hart in Nyx kodiert, da für die Werte ”Anzahl der MAC-Adressen” und ”Verkehrsanteil” keine harten Grenzen festgelegt werden sollen. Somit kann Nyx sehr flexibel trainiert werden. Ein zufälliger Ausschnitt dieser Daten wird als Seed (Trainingsdaten) verwendet. Der Trainingsdatensatz wird manuell annotiert[4] und dient dem Algorithmus zum Lernen des charakteristischen Musters eines Up-/Downlinkports. Nyx fragt zur Klassifizierung die Daten aller Ports aller Switches ab. Basierend auf den erlernten Erkenntnissen ist Nyx dann in der Lage, die Ports der Endgeräte zu erkennen, also Up-/Downlinkports von Datenports zu unterscheiden. Gespeichert werden aber nur die MAC-Adressen, die nicht von einem Up-/Downlinkport kommen. Weitere Details zur Realisierung des maschinellen Lernens sind in Abschnitt 5.2 zu finden.

4.2 Datenmodell: Zuordnung IP - MAC - Switch-Port

Ziel ist es sowohl die Ein- als auch die Ausgabe von Daten möglichst effizient zu gestalten. Beim Datenmodell nimmt die Relation IP-Adresse - Switch-Port eine zentrale Stellung ein. Sie entspricht dem Modell, das der Benutzer verwendet: er möchte zu einer bestimmten IP-Adresse den Switch und Switch-Port des zugehörigen Endgerätes erfahren.

Die Zuordnung IP-Adresse - Switch-Port erstreckt sich über Layer 2 (Sicherungsschicht) und 3 (Vermittlungsschicht, ”IP”). Monitoring-Systeme arbeiten auf Layer 3 und liefern die IP-Adresse des gesuchten Endgerätes, dem ein Edge-Switch-Port zugeordnet werden soll. Die MAC-Adresse (Layer 2) des Endgerätes ist zu diesem Zeitpunkt aber noch nicht bekannt. Diese Informationen liefert der zuständige Router des entsprechenden Subnetzes aus seiner ARP-Tabelle (Zuordnung IP - MAC). Switches liefern dann die Zuordnung MAC - Switch-Port (MAC-Forwarding-Tabelle). Diese Daten werden in getrennten Tabellen in einer Datenbank gespeichert:

- ARP-Tabelle: IP-MAC-Zuordnung
- MAC-Tabelle: MAC- Switch-ID - Switch-Port -Zuordnung

Sie werden erst bei der Auswertung bzw. Suche korreliert. Die Port-Bezeichnung eines Switches in einem Netzwerk ist nicht eindeutig. In der Regel werden die Ports pro Switch einfach durchnummeriert. Deshalb ist zusätzlich noch eine eindeutige Identifizierung des Switches notwendig. Bei Nyx dient die Seriennummer des Switches als Switch-ID. Somit entsteht eine eindeutige Verknüpfung zwischen einer MAC-Adresse und einem Switch-Port an einem bestimmten Switch.

In der Datenbank werden die Informationen aus den ARP-Tabellen der Router und den MAC-Forwarding-Tabellen der Switches außerdem mit einem Zeitstempel versehen. So entsteht eine History mit der Endgeräte im Nachhinein auch über mehrere Tage oder Wochen zurückverfolgbar sind.

Bei Lokalisierungsanfragen beginnend mit einer IP-Adresse werden die Daten beider Tabellen ausgewertet. Zuerst wird in der ARP-Tabelle die MAC-Adresse zur gesuchten IP-Adresse erfragt. Die MAC-Tabelle wurde durch die Topologieerkennung gefiltert, sodass

[4] von Hand markiert, welche Ports Up-/Downlinkports und welche Datenports in der Realität sind

im Idealfall (siehe Kapitel 6.2, "Probleme und Sonderfälle aus der Praxis") nur noch Daten aller Edge-Ports darin enthalten sind und somit zu jeder dem System bekannten MAC-Adresse die Information über den Anschlussport geliefert werden kann.

4.3 Parallelisierung

Nyx ist zur Verwendung in großen Netzen ausgelegt und erhebt den Anspruch, die Daten quasi in Echtzeit zur Verfügung zu stellen. Echtzeit bedeutet hier, dass die Daten im Zeitfenster der Aging-Time der MAC-Forwarding-Tabellen der Switches abgefragt werden müssen. Pro MAC-Adresse gibt es im Switch einen Aging-Timer. Läuft er auf Null wird die MAC-Adresse aus der Tabelle gelöscht, wenn sie in der Zwischenzeit keinen Verkehr mehr verursachte. Die Aging-Time von MAC-Adressen beträgt in der Regel fünf Minuten, bei IP-Adressen in der ARP-Tabelle sind es bis zu vier Stunden. Deshalb müssen alle Netzkomponenten innerhalb dieser Zeitfenster abgefragt werden, bevor wichtige Informationen verloren gehen.

In einem großen Netz mit mehreren hundert Switches würde eine serielle Abfrage aller Geräte viel zu lange dauern. Das Abfragen der einzelnen Netzkomponenten bei Nyx erfolgt deshalb mit mehreren parallelen Prozessen (Threads). Die maximale Anzahl der Threads ist konfigurierbar und wird der Leitsungsfähigkeit der Servers angepasst. Intern wird eine Warteschlange (Queue) der Netzkomponenten mit Zeitstempeln verwaltet, in der festgehalten wird, zu welchem Zeitpunkt eine Netzkomponenten zum letzten Mal abgefragt wurde. Jeder freie Thread fragt die nächste Netzkomponenten ab, die das 5-Minuten-Zeitfenster überschritten hat und gerade nicht von einem anderen Thread abgefragt wird, also für ihn blockiert ist. Somit wird sichergestellt, dass alle Netzkomponenten in Echtzeit abgefragt werden, der Server aber nicht überlastet wird.

5 Realisierung

Die am LRZ entwickelte Software Nyx ist modular aufgebaut und besteht aus folgenden Komponenten (vgl. Abbildung 2):

- einer externen MySQL-Datenbank, zur Implementierung des Datenmodells aus Abschnitt 4.2
- eine parallelisierte Komponente zur Datenakquisition, Filterung der relevanten Daten und Speicherung dieser Rohdaten (Crawler)
- eine parallelisierte Komponente zur Verdichtung der Rohdaten und Übertragung in die endgültigen Datenbanktabellen (Consolidator)
- eine parallelisierte Komponente zur Aufbereitung der Verkehrsdaten der Switches (Meta-Crawler)
- eine Komponente zur Topologieerkennung mit maschinellem Lernalgorithmus, zur Klassifizierung der Switch-Ports (Classifier)
- eine parallelisierte Komponente, die Serviceaufgaben (Hinzufügen von neuen Netzkomponenten, löschen alter Einträge, Datenbankbereinigung, ...) erledigt (Service-Controller)
- eine Komponente zur Interaktion/Bearbeiten von XML-Anfragen der webbasierten Benutzeroberfläche oder externen Programmen (I/O-Interface)

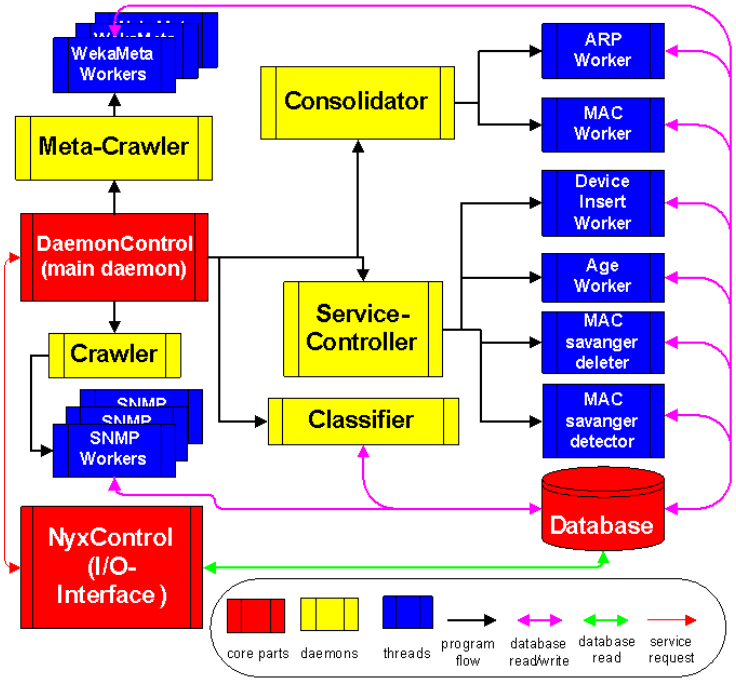

Abbildung 2: Nyx-Architektur

Die Rohdaten aller Netzkomponenten werden vom Crawler per Managementprotokoll SNMP im 5-Minutenmittel (siehe Abschnitt 4.3) abgefragt, nach Datenports gefiltert und in der Datenbank temporär zwischengespeichert. Dort werden sie vom Consolidator verdichtet und in der Datenbank in den endgültigen Tabellen (vgl. Datenmodell aus Abschnitt 4.2) permanent gespeichert. Die Erstellung der Verkehrsdaten erfolgt durch den Meta-Crawler ebenfalls im 5-Minutenintervall, indem ebenfalls per SNMP Daten von Netzkomponenten abgefragt werden. Die Topologieerkennungskomponente (Classifier) wird alle 30 Minuten aktiv. Sie enthält den maschinellen Lernalgorithmus, der basierend auf den Verkehrsdaten alle Switch-Ports klassifiziert. Der Service-Controller erledigt regelmäßige Aufgaben: er nimmt auf externen Befehl hin neue Netzkomponenten in die Datenbank auf (Device Insert Worker). Außerdem werden einmal pro Tag Einträge, die älter als 10 Tage sind, gelöscht (Age Worker) und die Datenbank bereinigt (MAC Scavanger Detector & Deleter). Das I/O-Interface dient zur Kommunikation mit externen Programmen über eine XML-Schnittstelle (siehe Abschnitt 5.1) sowie einer Weboberfläche für Benutzeranfragen.

5.1 Softwaredesign

Das objektorientierte Design, soll eine gute Wartbarkeit und Erweiterbarkeit von Nyx garantieren. Java wurde als Programmiersprache gewählt. Die Software ist in mehrere Daemons (Subprozesse) unterteilt. Jeweils ein Daemon realisiert eine der oben beschriebenen Komponenten. Die meisten Daemons arbeiten intern parallelisiert mit mehreren Threads. Durch dieses Konzept ist es ohne Probleme möglich, neue Aufgaben als Daemons zu implementieren. Da alle Daemons praktisch unabhängig voneinander arbeiten, können zu Test- und Debugging-Zwecken Daemons auch einzeln ausgeführt werden. Prinzipiell wäre es sogar möglich, jeden Daemon auf einem eigenen Server zu betreiben.

Netzkomponenten sind in Nyx als universell erweiterbare Objekte repräsentiert. Daraus wird dann für einen Netzkomponententyp eine eigene Geräteabstraktion abgeleitet, in der die gerätespezifischen Object Identifier (OIDs) für SNMP und andere Parameter, wie z.B. der Grundtyp (Layer 2 oder 3 Gerät), festgelegt werden. Nyx ist so praktisch auf jede SNMP-fähige Netzkomponente erweiterbar, die SNMP-MIBs unterstützen. Durch das objektorientierte Klassendesign der Geräteabstraktionen können von ihnen auch weitere Subtypen, d.h. speziellere Gerätebeschreibungen abgeleitet werden. Beim LRZ werden durch Nyx verschiedene Gerätetypen von Cisco Systems, Hewlett Packard und F5 Networks per SNMP abgefragt.

Zur Kommunikation nach Außen wurde eine universell einsetzbare XML-Schnittstelle implementiert. Nyx kann so von externen Anwendungen leicht abgefragt werden. Auch die Benutzeroberfläche kommuniziert mit dem Nyx-Backend über diese XML-Schnittstelle.

Die Grundlagen des Softwaredesigns wurden im Rahmen eines studentischen Praktikums ausgearbeitet [Fis05]. Bei dem hier vorgestellten System handelt es daher um Erweiterungen und Verbesserungen.

5.2 Maschinelles Lernen

Maschinelles Lernen als Form des Data Mining eignet sich vor allem dann gut, wenn komplexe, chaotische Systeme betrachtet werden.

Nyx arbeitet mit maschinellen Lernen, um möglichst effizient Up-/Downlinkports erkennen zu können. Dabei wird die Java-Bibliothek WEKA [Pro] verwendet. Als Lernalgorithmus kommt J48 zum Einsatz, der zum Lernen intern einen Entscheidungsbaum (C4.5 decision tree [Qui93]) aufbaut. Dabei wird dem Algorithmus ein kleiner Datensatz (1000 von 50000 Ports) vorgegeben (”supervised learning”, [uDGS04]), bei dem Up-/Downlinkports manuell oder per Skript annotiert wurden. Daraus wird dann der Entscheidungsbaum aufgebaut, indem die Datensätze der einzelnen Ports den Basisklassen ”Up-/Downlinkport” oder ”Datenport”, abhängig von der Annotation, über ein Wahrscheinlichkeitsmodell zugeordnet werden. Hierbei handelt es sich um statisches Lernen. Der Entscheidungsbaum wird einmal beim Start von Nyx aufgebaut und bleibt während des gesamten Betriebs unverändert. Beim späteren Klassifizieren der 50000 Ports werden pro Port dessen aktuelle Verkehrsdaten an den Algorithmus weitergeleitet. Auf dieser Basis wird mit dem zu Beginn gelerntem Entscheidungsbaum berechnet, ob es sich um einen Up-/Downlinkport handelt.

Der Vorteil des maschinellen Lernens gegenüber dem konventionellen Erstellen von Erkennungsregeln ist, dass maschinelles Lernen gut mit ”unscharfen” Daten umgehen kann. Zum Beispiel kann man keine konkreten Grenzen angeben, ab wie vielen MAC-Adressen, die man an einem Port sieht, es sich um einen Up-/Downlinkport handelt. Sind es nur ein oder zwei, ist es mit hoher Wahrscheinlichkeit ein Datenport. Sind es mehr als 2, kann es sowohl ein Up-/Downlinkport sein, ein Server mit mehreren MAC-Adressen, z.B durch Virtualisierungslösungen, oder aber ein nicht verwaltbarer Mini-Switch, der deshalb als Endgerät behandelt wird.

6 Erfahrungen aus dem praktischen Betrieb

Nyx befindet sich seit Januar 2007 im MWN im produktiven Betrieb. Im folgenden werden Details erläuert, die beim praktischen Einsatz relevant sind, sowie Probleme und Sonderfälle, die dabei zu Tage traten.

6.1 Hardwareanforderungen

Wegen der großen Last, die schon während der Entwicklungsphase deutlich wurde, wird Nyx im LRZ auf zwei getrennten Servern betrieben. Auf dem ersten Rechner wird ein Java-Tomcat-Server mit der Nyx-Software ausgeführt. Es handelt sich dabei um eine Intel Dual-Xeon Maschine mit je 2,8 GHz mit Hyperthreading und 4 GB Arbeitsspeicher. Der zweite Server ist der Datenbankserver, der zwei AMD Opteron-Prozessoren mit je 2,6 GHz und 4 GB RAM enthält. Als Betriebssystem wird SuSE Linux verwendet. Mit einer durchschnittlichen CPU-Last von 40-50% (siehe Abbildung 3) bzw. 80 % sind beide Server gut ausgelastet. Die Netzwerklast beider Server beträgt nahezu konstant 28 Mbit/s, wobei dieser Verkehr hauptsächlich durch die Kommunikation zwischen der Nyx-Software und der Datenbank erzeugt wird, wenn die Datenbank Daten an Nyx zurückliefert. Die andauernden SNMP-Abfragen der Netzkomponenten (1-2 Mbit/s) fallen dabei kaum ins Gewicht.

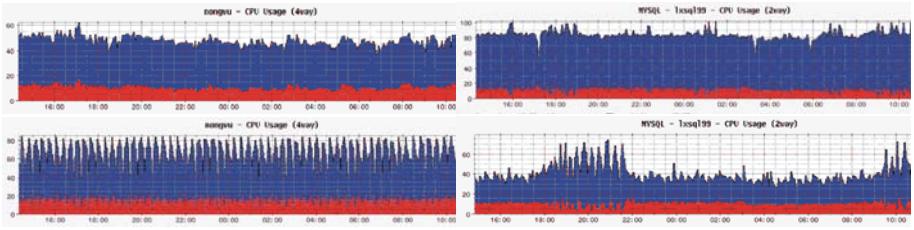

Abbildung 3: CPU-Tagesauslastung des Nyx- und Datenbankservers vor der Optimierung (oben) und danach

Nachdem es im Mai 2007 zu Engpässen beim Datenbankserver kam (CPU-Last fast 100% über längere Zeiträume) und dadurch die Verarbeitung und Verdichtung der Rohdaten mit dem Consolidator verzögert wurde, war eine Optimierung notwendig. Dabei wurden in der Meta-Crawler-Komponente komplexe SQL-Statements so verändert und vereinfacht, dass die Hauptlast nicht mehr vom Datenbankserver, sondern von Nyx selbst in Java verarbeitet wird.

Danach zeigte sich eine deutliche Veränderung: wie erwartet stieg die CPU-Last des Nyx-Servers (auf 60-80%) und die des Datenbankservers ging zurück, auf 40-60%. Außerdem ist nun im 5-Minuten-Mittel keine nahezu konstante CPU-Auslastung mehr zu beobachten (vor allem beim Nyx-Server), sondern eine ausgepägte Kammstruktur. Das ist eindeutig auf die Optimierungen zurückzuführen: die veränderten Aufgaben im Meta-Crawler wurden auch sinnvoll zeitlich optimiert, sodass sie nicht mehr pro Netzkomponente, sondern nur noch alle 15 Minuten ausgeführt werden. Auch der Datenbankserver wird nicht mehr konstant belastet und zeigt daher ausgeprägte, aber weniger konstante Lastspitzen.

6.2 Probleme und Sonderfälle aus der Praxis

Schon während der ersten Testphasen von Nyx zeigte sich, dass in der Praxis neben der Serverbelastung andere Probleme und Sonderfälle auftreten, die beim theoretischen Entwurf von Nyx keine große Rolle spielten.

Nicht erkannte Up-/Downlinks (Up-/Downlink als Datenport erkannt) führen zu überflüssigen Daten. MAC-Adressen sind dann am Up-/Downlink, am eigentlichen Datenport und ggf. und weiteren Ports sichtbar, z.B. bei Bridging-Firewalls, deren beide Ports am selben Switch angeschlossen sind (jedoch in unterschiedlichen VLANs). Für Nyx entsteht dann der Eindruck, als ob das Endgerät mit der gesuchten MAC-Adresse immer zwischen diesen zwei Ports hin und her springen würde. Pro Sprung erfolgt ein neuer mit Zeitstempel versehener Eintrag in der Datenbank. In einer solchen Situation ist es sehr schwierig, den ei-

gentlichen Datenport zu finden. Da einer oder mehrere Up-/Downlinkports als Datenports erkannt wurden, lässt sich an Hand der Verkehrsdaten nicht auf den richtigen Datenport schließen. In solchen Fällen kann eine Veränderung bzw. Verbesserung der Trainingsdaten Abhilfe schaffen. Ansonsten ist eine manuelle Recherche mit anschließender Korrektur notwendig, was aber sehr zeitaufwändig und nur im Einzelfall durchführbar ist. Da es im MWN eine größere Anzahl Bridging-Firewalls gibt, wurde ein zusätzlicher Filter in die Crawler-Komponente (Datenakquisition) eingebaut. Er erkennt, wenn eine MAC-Adresse an einem Switch an mehreren Ports in unterschiedlichen VLANs auftaucht. Anhand einer manuell gepflegten Liste wird erkannt, welches das interne VLAN hinter der Firewall ist. Die MAC-Adressen des anderen Ports, also der Uplink der Firewall, werden gefiltert und der Port als Up-/Downlink für die Zukunft klassifiziert.

Um nicht die Datenbank mit Springer-MACs, die der Bridging-Firewall-Filter nicht erkennen kann, zu überfluten (das können mehrere zehntausend Einträge pro Tag sein!), wurde ein zusätzlicher Service-Prozess implementiert (MAC Scavanger Detector & Deleter in Abbildung 2). Er erkennt alle 60 Minuten die Springer-MACs, die mehr als 100 Datenbankeinträge haben und übernimmt ihre MAC-Adressen in eine eigene Tabelle. Einmal pro Tag werden dann die MAC-Adressen, die in dieser Tabelle stehen, aus der History-MAC-Tabelle der Datenbank gelöscht.

7 Zusammenfassung und Ausblick

Das Nyx-Projekt am LRZ beschäftigt sich mit der Lokalisierung von Endgeräten in großen heterogenen Netzwerken. Um von einer IP-Adresse, die beim Security-Management auffällig geworden ist, zu einem bestimmten Endgerät bzw. einem Port auf einem Edge-Switch zu kommen, ist ein nicht unerheblicher Aufwand erforderlich. Dieser wird durch die Automatisierung innerhalb von Nyx deutlich reduziert.

Zuerst werden alle Daten der Netzkomponenten zentral gesammelt, gefiltert und verdichtet. Dazu werden aber Informationen über die Topologie des Netzes benötigt. Setzt man heterogene Netzkomponenten ein, sind aber standardisierte Protokolle zur Topologieerkennung genauso wenig erfolgreich wie einfache heuristische Verfahren. Daher kommt ein maschineller Lernalgorithmus zu Einsatz. Nyx ist deshalb in der Lage mit flexiblen Kriterien Up-/Downlinkports von Datenports zu unterscheiden. Die Qualität dieser Topologieerkennung hängt aber sehr stark von den Trainingsdaten ab, sodass auch bei guten Trainingsdaten einzelne Ports falsch erkannt werden können. Deshalb müssen ggf. zusätzliche Filter verwendet werden, um die durch nicht erkannte Up-/Downlinkports entstehende Datenflut einzudämmen.

Nyx wird am LRZ kontinuierlich weiterentwickelt und soll im Sommer 2007 als Open-Source-Projekt veröffentlicht werden. Bei der Weiterentwicklung werden die Trainingsdaten immer weiter verfeinert und an die Anforderungen der Praxis angepasst werden. Weiterhin sollen neue Geräte, z.B. Access-Points zur Lokalisierung von drahtlosen Endgeräten (Access Point statt Switch-Port), in Nyx aufgenommen werden. Auch die zusätzlichen Filter werden weiterentwickelt, um z.B. Bridging-Firewalls noch besser erkennen zu können. Durch die eventuelle Hinzunahme weiterer Verkehrsdatenkriterien und ggf. Verwendung anderer Lernalgorithmen der WEKA-Bibliothek könnte die Erkennungsleistung von Nyx weiter verbessert werden, damit die Suche nach einer IP-Adresse keine Suche nach der Nadel im Heuhaufen bleibt.

Literatur

[BGJ+04] Yuri Breitbart, Minos Garofalakis, Ben Jai, Cliff Martin, Rajeev Rastogi, and Avi Silberschatz. Topology discovery in heterogeneous IP networks: the NetInventory system. *IEEE/ACM Trans. Netw.*, 12(3):401–414, 2004.

[Fis05] S. Fischer. Echtzeitanalyse von IP- und MAC–Adressen auf den Netzkomponenten des MWN als Webservice. Technical report, March 2005.

[IEE05] IEEE. 802.1AB-2005: Station and Media Access Control Connectivity Discovery. Technical report, 2005. `http://standards.ieee.org/getieee802/download/802.1AB-2005.pdf`.

[LR06] Leibniz-Rechenzentrum. MWN Netzkonzept. Technical report, 2006. `http://www.lrz-muenchen.de/services/netz/mwn-netzkonzept/mwn-netzkonzep%t.pdf`.

[MFB] Eric Miller, Bill Fenner, and Max Baker. Netdisco. `http://netdisco.org/`.

[Pro] WEKA Machine Learning Project. `http://www.cs.waikato.ac.nz/~ml/weka/`.

[Qui93] J. Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[Sys] Cisco Systems. Configuring Cisco Discovery Protocol. `http://cisco.com/univercd/cc/td/doc/product/software/ios122/122cgcr/ffu%n_c/fcfprt3/fcf015.htm`.

[uDGS04] Dr. S. Staab und Dr. Gerd Stumme. Vorlesung Knowledge Discovery, WS 2003/2004, Kapitel VI.1 "überwachte Data und Data Mining Verfahren". Technical report, Universität Karlsruhe, 2004. `http://www.aifb.uni-karlsruhe.de/Lehre/Winter2003-04/kdd/download.htm`.